REMARKS

Claims 1-31 are pending in the application. Claims 32-95 were cancelled per the restriction requirement in the previous office action response. Claims 1, 3-10, and 12-31 have been amended.

Objection to the Drawings:

Applicant submits that the amendment to Fig. 23 overcomes the objection thereto.

Objection to the Claims:

Applicant has amended the claims to remove the reference characters and the quotation marks. In light of these amendments, Applicant submits that the objection has been overcome.

35 U.S.C. § 112 Rejections:

In light of the amendment to the specification and Fig. 23, Applicant submits that the claims are in compliance with 35 U.S.C. § 112. Applicant notes that instruction fetch unit 203 includes the program counter (pc).

35 U.S.C. § 102 and §103 Rejections:

Claims 1-31 were rejected under 35 U.S.C. § 102(b) as being anticipated by Luke, U.S. Patent 6,505,267. Applicant respectfully traverses this rejection.

The cited reference fails to teach or suggest all of the elements of the independent claims. Luke teaches "'[a] Universal Serial Bus to parallel bus bridge [including] a Universal Serial Bus port that receives a serial bit stream of data and commands in a Universal Serial Bus protocol from a USB host computer. A parallel bus port on the bridge includes parallel port registers and state machines coupled to a peripheral device. A USB controller core is coupled between the Universal Serial Bus port and the parallel bus port and converts data and commands between the Universal Serial Bus protocol and the parallel bus protocol. A sequencer is coupled between the

USB controller core and the parallel bus port. A sequence of sequencer commands is loaded into memory in the USB bridge and used by the sequencer to perform a sequence of parallel port operations. The sequencer performs the commands autonomously without intervention from the USB host computer. Because the host computer does not have to initiate a USB transaction for each individual parallel port operation, the sequence of operations is completed in a shorter amount of time." (Abstract, Luke).

Independent claim 1 recites:

"An SMBus message handler comprising:

a memory configured to store **microcode** comprising at least two programs each for handling a bus command protocol and comprising at least one instruction;

an interface to a register configured to identify a starting address of a program in said memory;

an instruction fetch unit configured to **read** an instruction at an address in said memory , said address being specified by a program counter; and

a finite-state machine configured to receive **and interpret** the instructions read by said instruction fetch unit and manage the data transfer between an **SMBus** interface and a register set in compliance with said instructions read from said memory." (Emphasis added)

Independent claims 10 and 19 recite similar combinations of features.

In the office action, the Examiner contends that Luke teaches a memory configured to store microcode comprising at least two programs, citing Fig. 2, elements 32 and 40, and col. 7, line 58 to col. 9, line 12. Applicant respectfully disagrees with the Examiner's characterization. Nowhere in the citation provided by the Examiner (or

elsewhere in Luke, for that matter) is there any teaching or suggestion of storing microcode. In fact, in col. 7, lines 66-67, Luke states

> The RRMW command functionality would look like the following in the **C** programming language: "RegisterValue=((RegisterValue & (.about.MaskField)).vertline.(DataField & MaskField))" (Emphasis added).

The C programming language to which Luke refers is a high-level programming language. In contrast, a micorpogram implements a CPU instruction set, with microprograms often being referred to as microcode. Attached herewith is a Wikipedia article directed to Microcode. **In light of the differences between microcode and high-level programming languages (such as the C programming language), Applicant submits that Luke fails to teach or suggest "a memory configured to store *microcode* comprising at least two programs" as recited in combination with the other features of claim 1 and similarly recited in the other independent claims.**

The Examiner further contends that element 90 of Luke's Fig. 6 reads on Applicant's recited instruction fetch unit. The Examiner also cites Luke in col. 7, lines 17-19, which states:

> "An op code latch 90 stores the operational code for the present sequencer command addressed by the program counter 84."

However, Applicant's independent claim 1 recites "an instruction fetch unit configured to **read** an instruction at an address in said memory, said address being specified by a program counter" (emphasis added). Nowhere in the above citation or elsewhere in Luke is there any teaching or suggestion that op code latch 90 is configured to **read** an instruction. **Accordingly, Luke's op code latch 90 does not read on Applicant's recited instruction fetch unit.**

The Examiner reads element 82 of Luke's Fig. 6 onto Applicant's recited "finite-state machine configured to receive **and interpret** the instructions read by said instruction fetch unit and manage the data transfer between an **SMBus** interface and a register set in compliance with said instructions read from said memory" (emphasis added). In support of this, the Examiner cites Luke at col. 7, lines 5-16 and lines 28-36, which state:

> "FIG. 6 is a detailed diagram of the sequencer 46. A state machine 82 receives signals from the USB interface 42 and controls a group of latches, program counters and status flags. Status flags 83 contain status information about the sequencer operations. A program counter 84 is coupled to the sequencer RAM 40 to sequence through address locations loaded with a sequencer program. A main loop counter 86 receives a value from the USB interface 42 which identifies the number of times the entire sequence is to be performed. An instruction loop counter 88 receives a loop count value from the sequencer RAM 40 that identifies the number of times the current command is to be performed." (Luke, col. 7, lines 5-16; emphasis added).

> "Control signals 102 are used by the state machine 82 to configure registers in the parallel port 44 for read or write operations. Control signals 104 are used by the state machine 82 to control when and how data is gated between the peripheral device 20 and the bulk_in and bulk_out buffers through the parallel port 44. The sequencer configures the parallel port 44 for these different operations by sending the appropriate signals to the parallel port state machines (FIG. 3)." (Luke, col. 7, lines 28-36; emphasis added).

Nowhere in the above citations, or elsewhere in Luke, is there any teaching or suggestion that state machine 82 **interprets** instructions. Furthermore, nowhere in the above citations or elsewhere in Luke is there any teaching or suggestion that state machine 82

"[**manages] the data transfer** between an **SMBus** interface and a register set in compliance with said instructions read from said memory" as recited in combination with the other features of claim 1 and similarly recited in the other independent claims. In fact, since Luke is directed to a USB peripheral bridge and does not provide any disclosure of an SMBus, it follows that Luke does not teach any functionality regarding data transfer between an **SMBus** interface and any other location. **Accordingly, Luke fails to teach or suggest a "finite-state machine configured to receive and *interpret* the instructions read by said instruction fetch unit and *manage the data transfer* between an *SMBus* interface and a register set in compliance with said instructions read from said memory " as recited in combination with the other features of claim 1 and similarly recited in the other independent claims.**

Finally, while the Examiner contends that Luke discloses a System Management Bus (SMBus) message handler, Applicant notes that Luke is directed to a Universal Serial Bus (USB) peripheral bridge. USB is an entirely different bus than SMBus. Luke makes no mention whatsoever of an SMBus, and therefore fails to anticipate the independent claims.

Thus, for at least the reasons stated above, Applicant submits that Luke fails to teach or suggest all of the elements of the independent claim. Accordingly, removal of the 35 U.S.C. § 102(b) rejection is respectfully requested.

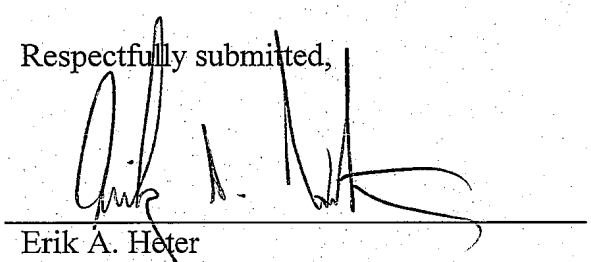
With regard to the § 103(a) rejections, Applicant notes that Luke is the primary reference for each. Thus, for at least the reasons stated above, Applicant submits that the cited references, taken singly or in combination, fail to teach or suggest all of the elements of the independent claims, as none of the secondary references remedy the deficiencies of Luke noted above. Accordingly, removal of the 35 U.S.C. § 103(a) rejection is respectfully requested.

CONCLUSION

Applicant submits the application is in condition for allowance, and an early notice to that effect is requested.

If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5500-92201/EAH.

Respectfully submitted,



Erik A. Heter
Reg. No. 50,652
AGENT FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800

Date: 7/9/2007